# *Agenda*

- How HTTP works?
- How HTML works?
- Cross-site scripting (XSS)
- Session management

# *How HTTP works?*

TCP connection 123.123.123.123:X -> 194.36.162.154:80

**GET /arhiiv.php?id=112108 HTTP/1.1**
Host: www.err.ee
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.11) Firefox/3.6.11
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive

194.36.162.154:80 -> 123.123.123.123:X

**HTTP/1.1 200 OK**
Content-Type: text/html; charset=utf-8
Server: Microsoft-IIS/7.5
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Sun, 28 Nov 2010 11:48:17 GMT
Content-Length: 19438

<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
ERR – Eesti Rahvusringhääling
</title>
[...]

# *How HTTP works?*

**POST /index.php?06219912&com=1 HTTP/1.1**
Host: uudised.err.ee
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.11) Firefox/3.6.11
Connection: keep-alive
Referer: http://uudised.err.ee/?06219912&com=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 118

commentpost_comment=This+is+test+comment&commentpost_replyto=0&formaction=postmaincomment&commentpost_displayname=John
**HTTP/1.1 200 OK**
Date: Sun, 28 Nov 2010 12:08:16 GMT
Server: Apache/1.3.26 (Unix) Debian GNU/Linux PHP/4.4.0-0.la02.0
X-Powered-By: PHP/4.4.0-0.la02.0
Cache-Control: no-cache
Pragma: no-cache
Connection: Keep-Alive
Content-Type: text/html; charset=utf-8

<html>
  <body>
[...]

# *How HTTP works?*

**POST /index.php?06219912&com=1 HTTP/1.1**
Host: uudised.err.ee
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.2.11) Firefox/3.6.11
Connection: keep-alive
Referer: http://uudised.err.ee/?06219912&com=1
Content-Type: application/x-www-form-urlencoded
Content-Length: 118

commentpost_comment=This+is+test+comment&commentpost_replyto=0&formaction=postmaincomment&commentpost_displayname=John

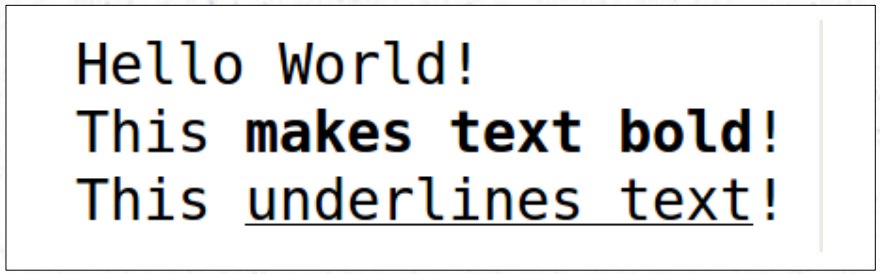
```
$_GET => Array
  (
     [06219912] =>
     [com] => 1
  )

$_POST => Array
  (
     [commentpost_comment] => This is test comment
     [commentpost_replyto] => 0
     [formaction] => postmaincomment
     [commentpost_displayname] => John
  )
```

```
$_SERVER => Array
  (
     [HTTP_HOST] => uudised.err.ee
     [HTTP_USER_AGENT] => Mozilla ...
     [HTTP_REFERER] => http://uudised...
     ...
  )
```

# *HTML tags*

```
<html>
  <body>
    <pre>
        Hello World!
        This <b>makes text bold</b>!
        This <u>underlines text</u>!
    </pre>
  </body>
</html>
```



# *How to display this?*

```
Please use <b> and </b> tag!
```

# *How to display angle brackets then?*

```
<html>
  <body>
    <pre>
        Please use <b> and </b> tag!
    </pre>
  </body>
</html>
```

Please use **and** tag!

# *HTML Entities*

| Character | Entity Number | Entity Name |
|---|---|---|
| " | &#34; | &quot; |
| ' | &#39; | ' (does not work in IE) |
| & | &#38; | &amp; |
| < | &#60; | &lt; |
| > | &#62; | &gt; |

# *HTML*

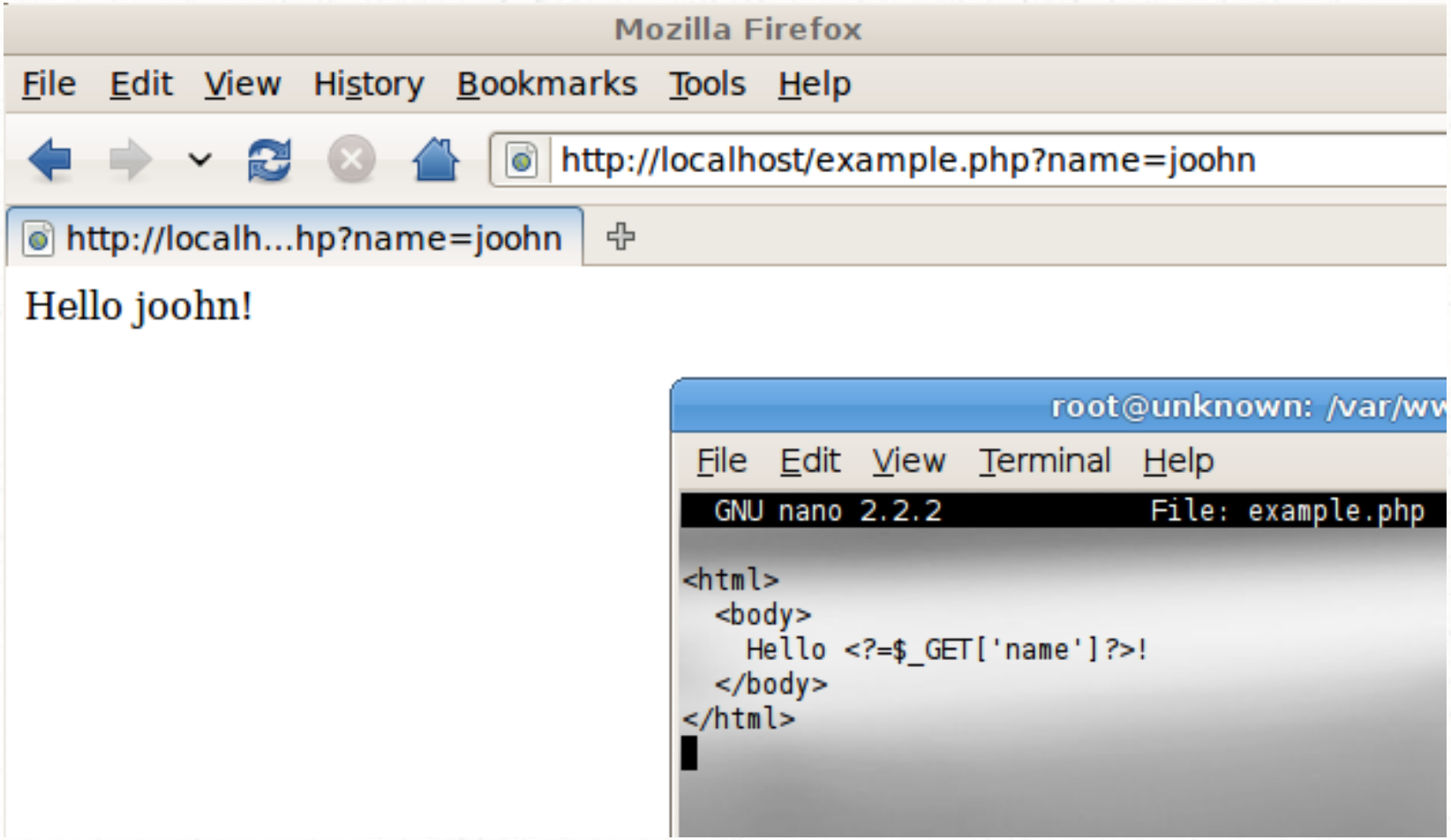
```
<html>
  <body>
    <pre>
       Please use &lt;b&gt; and &lt;/b&gt; tag!
    </pre>
  </body>
</html>
```

Please use  and  tag!

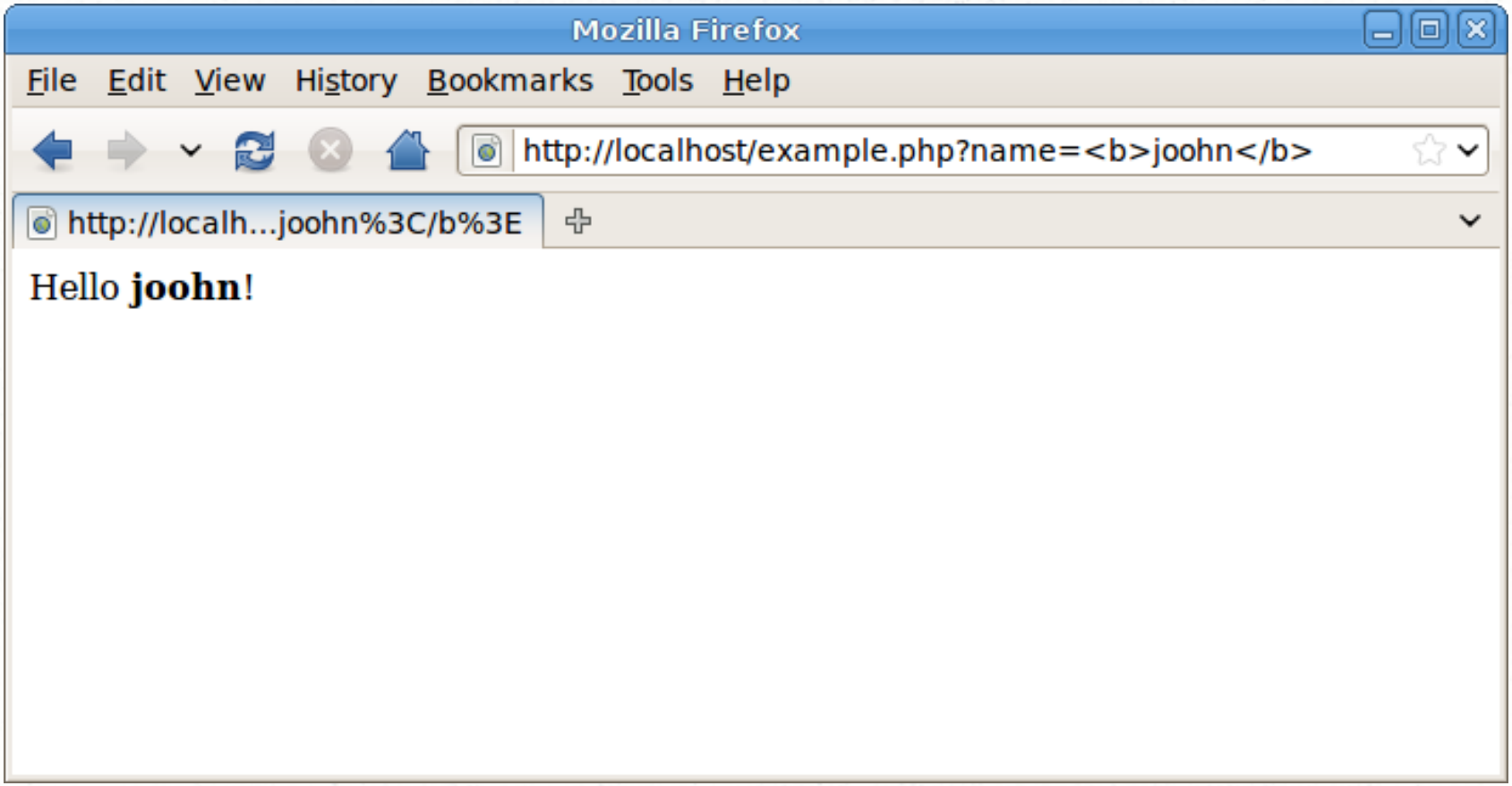
# *Cross-site scripting (XSS)*

Untrusted data used without proper escaping to dynamically generate HTML and show it to end-user (victim).

# XSS - example


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/example.php?name=joohn
http://localh...hp?name=joohn
Hello joohn!
root@unknown: /var/ww
File Edit View Terminal Help
GNU nano 2.2.2 File: example.php
<html>
<body>
Hello <?=$_GET['name']?>!
</body>
</html>

# XSS - example


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/example.php?name=joohn
http://localh...joohn%3C/b%3E
Hello joohn!

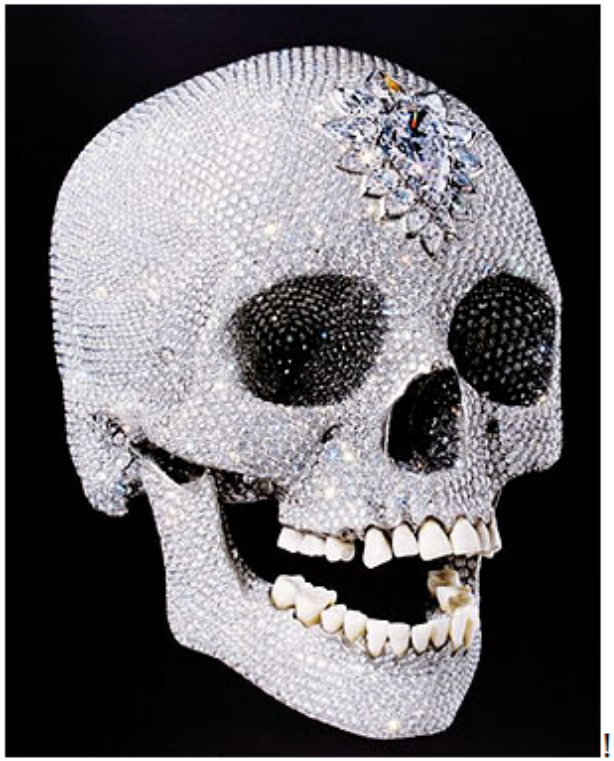
# XSS - example

Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost/example.php?name=<img src="http://www.artnewsblog.com/images/diamond-skull-2.jpg">

http://localho...ll-2.jpg%22%3E



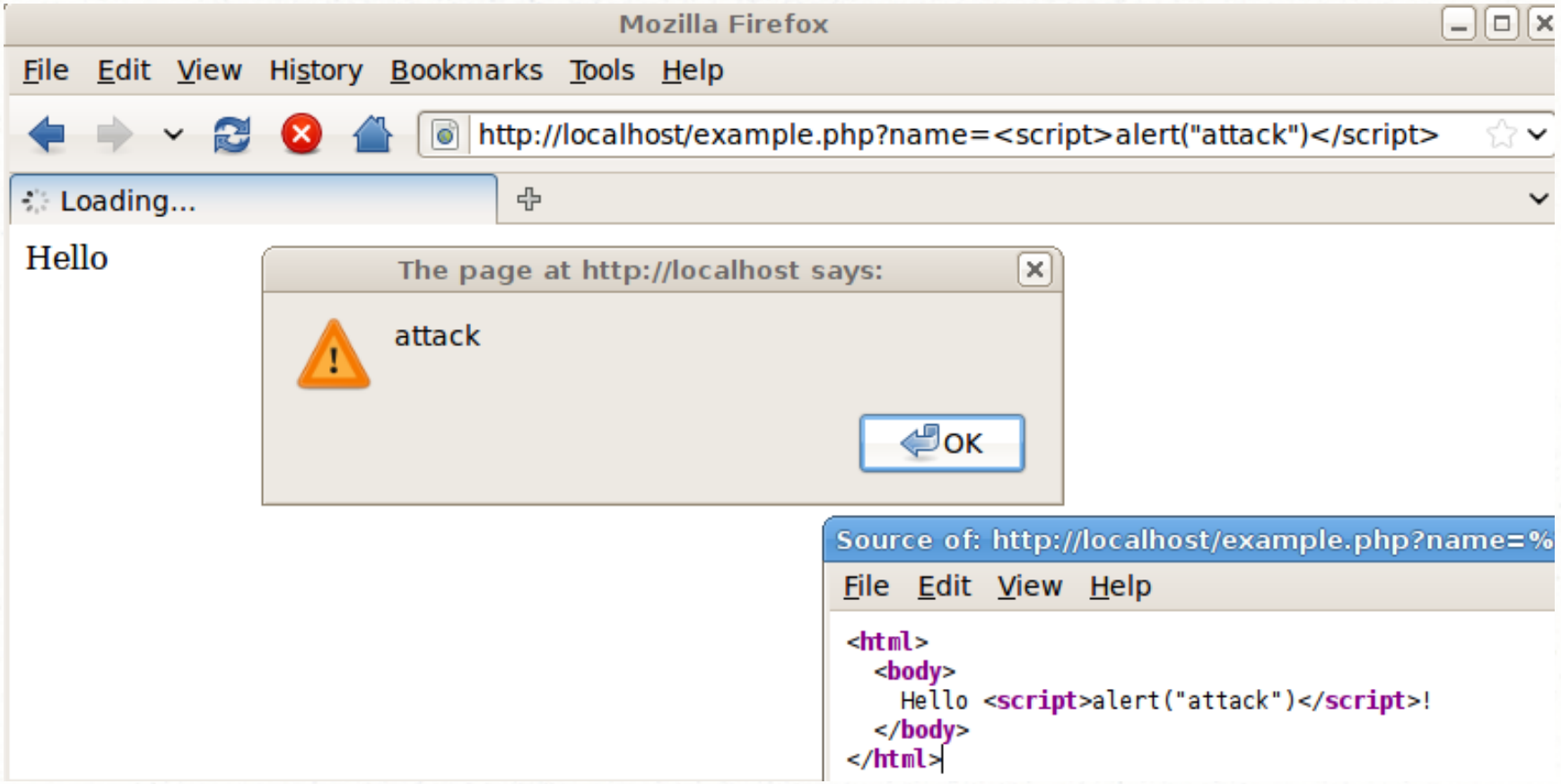
Hello !

Source of: http://localhost/example.php?name=%3Cimg%20src=%22http://w

File Edit View Help

```
<html>
  <body>
    Hello <img src="http://www.artnewsblog.com/images/diamond-skull-2.jpg">!
  </body>
</html>
```

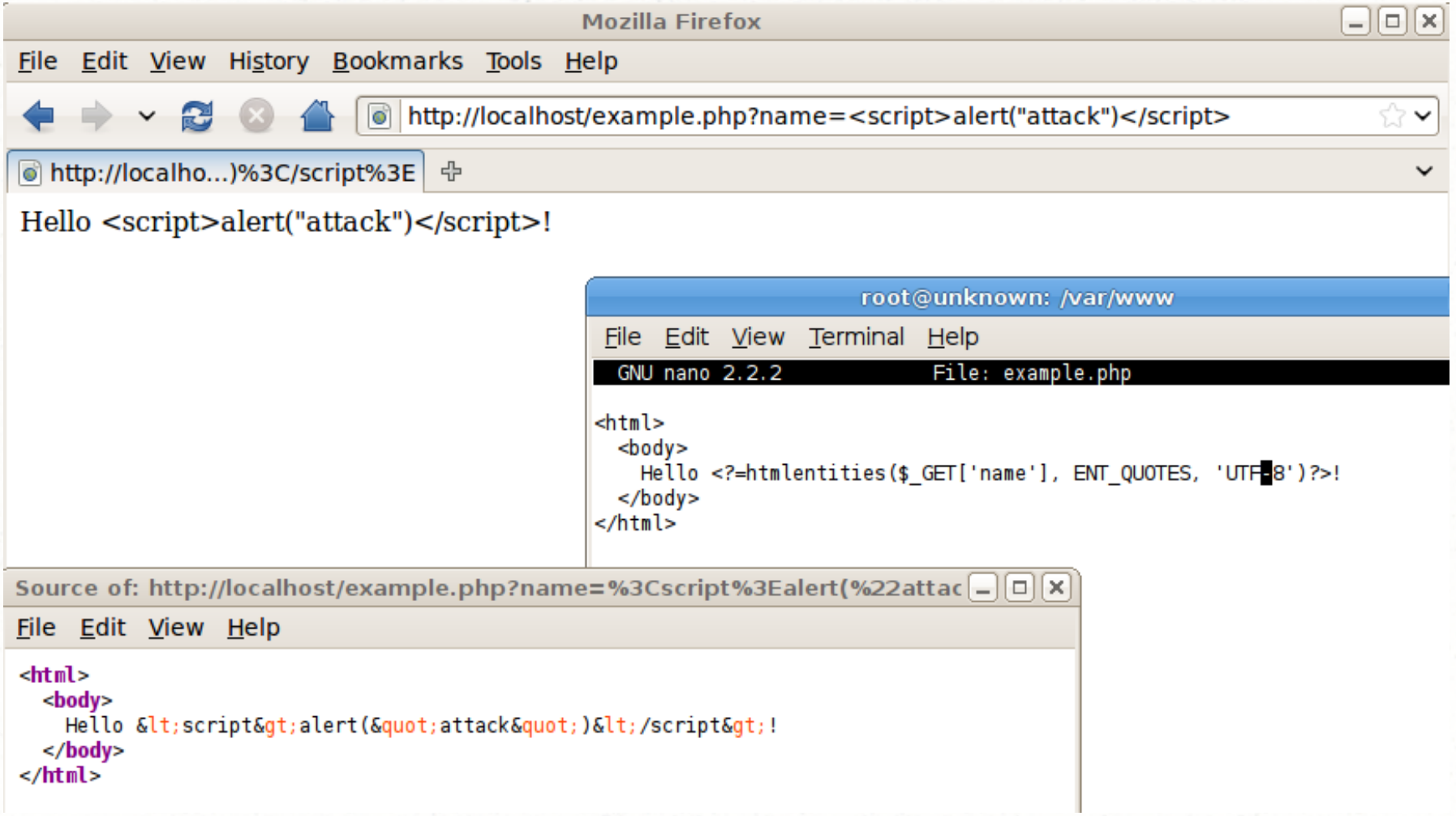
# XSS - example


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost/example.php?name=<script>alert("attack")</script>
Loading...
Hello
The page at http://localhost says:
attack
OK
Source of: http://localhost/example.php?name=%
File Edit View Help
<html>
<body>
Hello <script>alert("attack")</script>!
</body>
</html>

# *Types of XSS*

- Reflective XSS
  - Immediately used to generate page result (most common)
- Stored XSS
  - Stored on server and shown later when accessing "normal" URL's

# XSS - prevention (server side)



# *XSS prevention (client side)*

- Reflective only:
  - Internet Explorer 8 Cross-site scripting filter
  - Mozilla Firefox NoScript extension

# *XSS Summary*

- Most common web application flaw
- Client side attack
- Validate and escape user input!

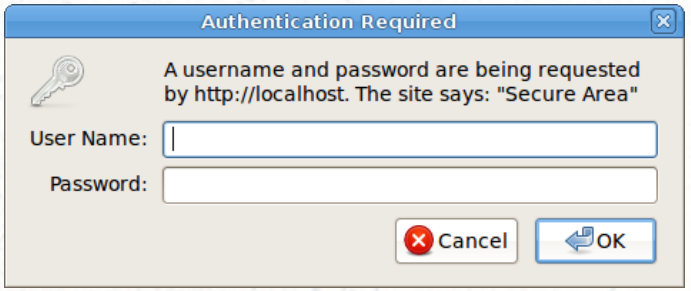
# *Session Management*

- HTTP is stateless protocol
- Several methods:
  - HTTP authentication
  - URL parameters
  - Cookies

# HTTP Basic Authentication

```
GET /private/index.html HTTP/1.1
Host: localhost
```

```
HTTP/1.1 401 Authorization Required
Server: HTTPd/1.0
Date: Sat, 27 Nov 2004 10:18:15 GMT
WWW-Authenticate: Basic realm="Secure Area"
Content-Type: text/html
Content-Length: 311
```




```
GET /private/index.html HTTP/1.1
Host: localhost
Authorization: Basic dXNlcm5hbWU6cGFzc3dvcmQK==
...
```

base64_encode('username:password') = "dXNlcm5hbWU6cGFzc3dvcmQK=="

# *HTTP URL Parameters*

- Session ID appended to links:

  http://www.example.com/index.php?SESSID=d1cea8b8c6b57afe

- Problems
  - HTTP Referers
  - Copy & Paste
  - Session Fixation

# *HTTP Cookies*

GET /index.php HTTP/1.1
Host: www.example.org

HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: SESSID=afe045bda8fc4

```
<html>
  <body>
    ...
```

```
$_COOKIES => Array
  (
    [SESSID] => afe045bda8fc4
    ...
  )
```

GET /logo.jpg HTTP/1.1
Host: www.example.org
Cookie: SESSID=afe045bda8fc4

# *HTTP Cookie Parameters*

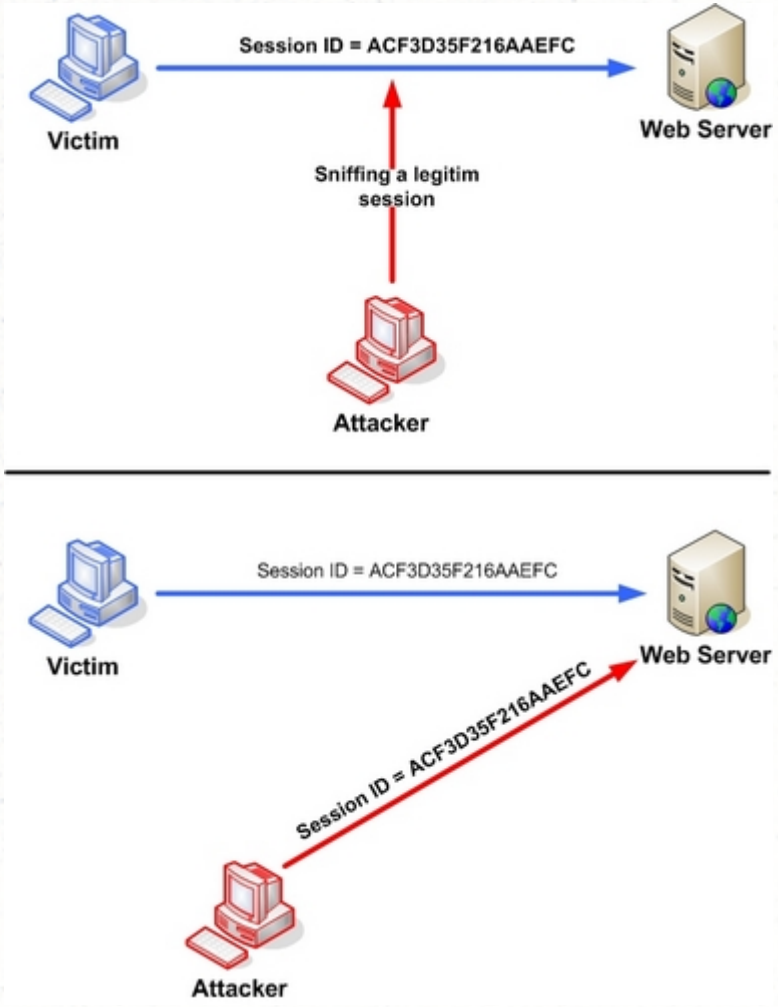
Set-Cookie: var=123abc; expires=Fri, 31-Dec-2010 23:59:59 GMT; path=/; domain=.example.net; HttpOnly; Secure;

# *Session Management - Attacks*

- Prediction
- Capture
- Fixation

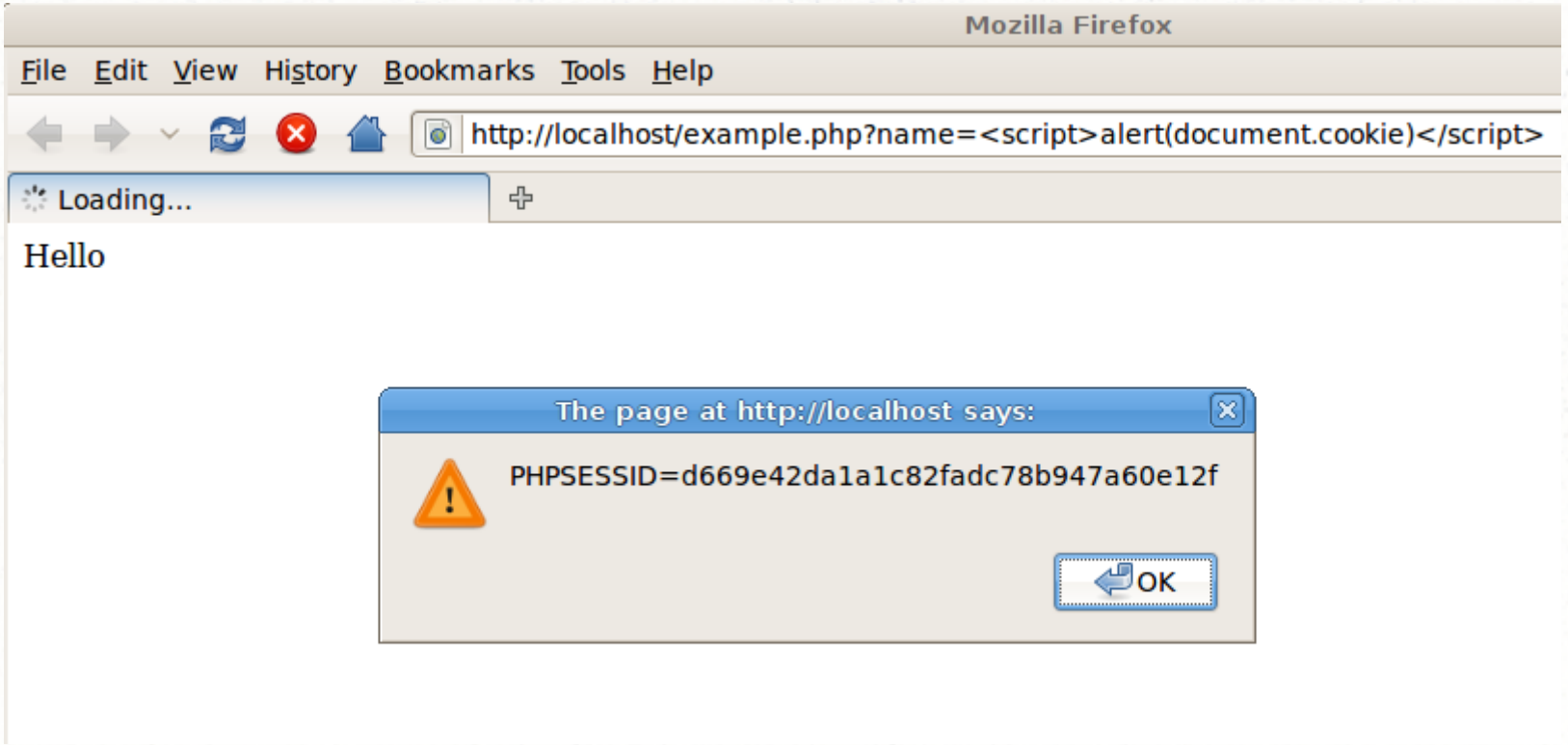
# *Session Capture - Sniffing*

- Sniffing unencrypted traffic
- HTTPS Cookie hijacking

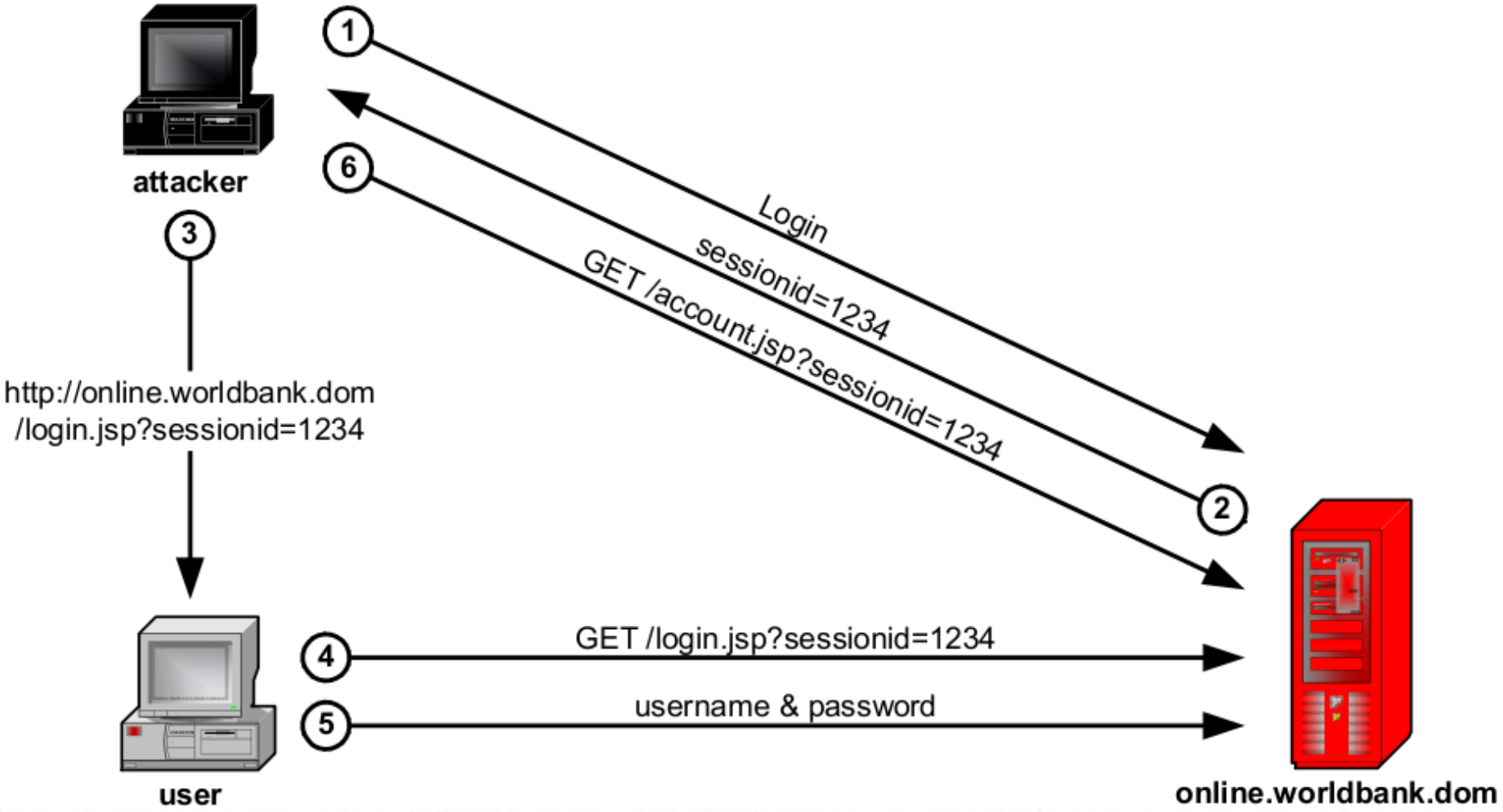


# *Session Capture - XSS*

- JavaScript can access cookies
  - Cookie 'HttpOnly' flag



# *Session Fixation Attack*



# *The devil lies in the details*

**browsersec**

*Browser Security Handbook*

- http://code.google.com/p/browsersec/